Quick Incompact3d guide and notes

This is a quick guide on to how to use **Incompact3d** and to know some details of its functioning. The present guide and the `modified` version of the code (baseline `v4.0`) are unofficial. Please refer to the official website of the code: https://www.incompact3d.com and the major references [1],[2],[3],[4].

Compiling

Compilation is first prepared with `cmake`, that generates the `make` file.
It is suggested to create a *build* directory in the same folder of the solver:

```
mkdir build
```

In the *build* directory run

```
cmake ../
```

After that, the *makefile* will be created. If a specific version of the Fortran compiler is needed, the option to be added is the following (e.g. `gfortran`):

```
cmake -DCMAKE_Fortran_COMPILER=gfortran ../
```

It is now possibly to compile the binary file in the build directory with the following command:

```
make -j n
```

where `n` is the number of processors that will be used to compile the program.

Basic functioning

In this section, the main features of the code that can be useful to run simulations are reported.

In Incompact3d:

1. Non-dimensional Navier-Stokes equations are solved.
2. The Re specified in the input file is used to compute the kinematic viscosity as

$$\nu = \frac{1}{Re}$$

so we are assuming unitary reference length and velocity scales. Reference scales depend then on the specific flow case.

Channel flow
For a channel flow, the default condition of simulation is to enforce a constant flow rate (CFR), while constant pressure gradient (CPG) can be enabled if necessary. Both conditions are rescaled with the centerline Reynolds number of a laminar Poiseuille flow, $Re_0 = \frac{U_0 h}{\nu}$. A relation is available in order to estimate the related friction Reynolds number Re_τ:

$$Re_\tau \approx 0.116 Re_0^{0.88}$$

With CPG option enabled, the same relation is used to impose the same non-dimensionalization, since Re_τ must be specified in the input file instead of Re_0:

$$Re_0 \approx \left(\frac{Re_\tau}{0.116} \right)^{1/0.88}$$

In CFR conditions, being U_0 constant and unitary, this means that the bulk velocity U_B must be kept to the value of $2/3$. This can be demonstrated being the laminar profile parabolic.

Finally, in order to estimate the related bulk Reynolds number $Re_B = \frac{2U_B h}{\nu}$, the following relation can be employed (Pope, "Turbulent Flows"):

$$Re_\tau \approx 0.09 Re_B^{0.88} \Rightarrow Re_B \approx \left(\frac{Re_\tau}{0.09} \right)^{1/0.88}$$

3. Pressure field is made non-dimensional with the reference specific kinetic energy content $q_{ref} = \frac{1}{2}\rho U_{ref}^2$, where U_{ref} is the reference velocity of the case. The code performs operation on pressure field with a factor Δt, but it is rescaled before saving the snapshots with `rescale_pressure` subroutine.
4. To evaluate the stretching parameter β for the mesh elements in y direction, some useful scripts can be found in the folder `1-pre_processing/stretching_mesh`. The default code of Incompact3d is `stretching_parameter_channel.f90` and it is used to setup channel flow simulations. For temporal TBL simulations, the Python script `mesh_evaluation_ttbl.py` was developed. This script can handle also pre-processing of channel flow cases. Low β values correspond to a strong stretching of the mesh, while high β values correspond to almost uniform mesh.
5. Variables are saved on y_p points, that are the faces of the grid elements, while y_{pi} points are the centers of the grid elements (i : internal).
6. Boundary values of velocity are specified through the b_{ijk} variables, where i is the wall-normal direction of the boundary, j is the direction of the specific velocity component and k specifies if we are considering the bottom or the top walls (e.g. b_{yx1} refers to the x velocity component, specified at the bottom boundary with normal direction y).
7. Boundary conditions are specified in the input file `input.i3d` with the following variables: `nclx1, nclxn, ncly1, nclyn, nclz1, nclzn`, that specify the normal direction to the boundary and if we are considering the first or the last element along the specific direction. Values that can be adopted are: 0, for *periodic BC*, 1 for *free-slip BC* and 2 for *Dirichlet BC* (so imposed velocity value). Boundary conditions can be different along the same dimension, so different combinations can be enforced.
8. Boundary conditions for scalar fields have the same functioning of the velocity BCs. They are called: `nclxS1, nclxSn, nclyS1, nclySn, nclzS1, nclzSn`.
9. The hyperviscous option for the second order derivative is a way to increase the numerical dissipation of the standard 6th order accurate scheme (that is sub-dissipative). In this manner, it is possible to increase the modified wavenumber at high wavenumbers, thus increasing the *dissipation error* E_{diss}, similarly to what is observed in high-order upwind schemes. This approach allows to prevent *wiggles* and thus to improve stability, even at high cell Reynolds number (or Péclet) $Pé \approx 200$. The two parameters available can be used also to have control on the numerical dissipation in the context of ILES simulations.
10. Implicit time integration is available for the diffusive terms in y direction. From preliminary analyses, it appears that it allows to drop the restriction due to the stability parameter $S < 1$ of fully explicit time integration schemes (Thompson et al. (1985)). The stability parameter S can be calculated in the three directions

$$S_i = \frac{u_i^2 \Delta t}{2\nu}$$

considering $i = x, y, z$.

11. Avoid to cancel the latest `restart.info` file, in order to being able to correctly calculate the current time unit (since the current time unit before restart is read from `restart.info` file itself, that is located in the `restart_info` folder).

12. The total shear velocity is calculated as:

$$u_\tau = \sqrt{\nu \frac{\partial U_{\parallel,w}}{\partial y}}$$

$U_\parallel = \sqrt{U^2 + W^2}$ and similarly for the x and z components: $u_{\tau,x} = \sqrt{\nu |\frac{\partial U_w}{\partial y}|}$ and $u_{\tau,z} = \sqrt{\nu |\frac{\partial W_w}{\partial y}|}$.

13. The friction coefficient c_f is calculated as:

$$c_f = 2\left(\frac{u_\tau}{U_{ref}}\right)^2$$

where U_{ref} is the reference velocity according to the specific flow case (U_w for a TTBL and U_B for a channel).

Visualization

Visualization of the results can be performed by opening the `.xdmf` files of the folder `data` with a visualization/post-processing software (e.g. Paraview). Moreover, it possible also to visualize 2D planes of the scalar field (if present), by opening `.xdmf` files that can be found inside the `data/planes` folder. This can be useful for large cases, being a single plane much less memory-demanding. The frequency of saving of the scalar planes can be set with the parameter `ioutput_plane` in the input file.

References

1. High-order compact schemes for incompressible flows: A simple and efficient method with quasi-spectral accuracy - Laizet & Lamballais - 2009
2. Incompact3d - A powerful tool to tackle turbulence problems with up to O(10^5) computational cores - Laizet & Li - 2011
3. Xcompact3D - An open-source framework for solving turbulence problems on a Cartesian mesh - Bartholomew et al. - 2020
4. Straightforward high-order numerical dissipation via the viscous term for DNS and LES - Lamballais et al. - 2011
5. Turbulent Flows - Pope
6. The cell Reynolds number myth - Thompson et al. - 1985